Sub-Account Architecture

Alpaca OmniSub Integration
Multi-Provider Omnibus Broker
KYC/KYB Onboarding Pipeline

Liquidity Inc.

Prepared by Zach Kelling, CTO of Satschel, Inc.

April 2026

Contents

1 Executive Summary

Liquidity.io operates as a registered Alternative Trading System (ATS), broker-dealer (BD), and transfer agent (TA) for digital securities. This document describes the sub-account architecture that underpins investor custody, order execution, and regulatory compliance across all three pillars.

The platform uses **Alpaca Securities** as the primary omnibus broker via their OmniSub model, with a multi-provider registry that supports 16 broker/exchange venues for smart order routing and liquidity aggregation.

Key design principles:

- Tenant isolation enforced at every layer (JWT → `X-Org-Id` header)
- Sub-account identity derived exclusively from server-side JWT validation
- Chain-first settlement: Liquid EVM is source of truth, SQL is cache
- Multi-provider: Alpaca primary, with failover to 15+ additional venues
- Regulatory compliance: KYC/AML gating before any account activation

2 Three-Pillar Architecture

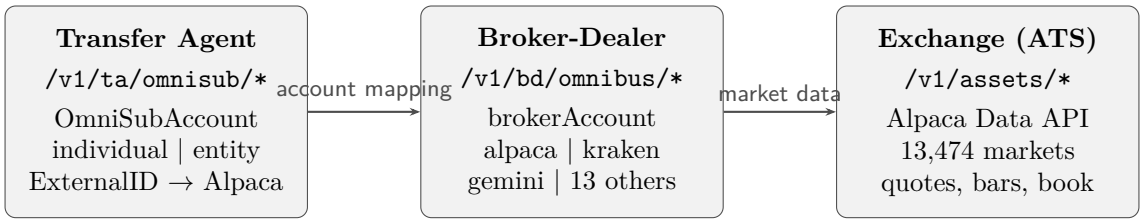

Figure 1: Three-pillar service architecture with sub-account flow.

All three services are single Go binaries powered by Liquid Platform (embedded SQLite, per-tenant encryption). They share a common auth model:

1. **Ingress** terminates TLS
2. **Gateway** validates JWT via JWKS, injects `X-User-Id` and `X-Org-Id`
3. **Service** strips all identity headers, re-derives from JWT, sets canonical headers
4. All queries scoped to `org_id` from the verified JWT—never from client input

2.1 Request Flow

Figure 2: Request flow: client to chain.

3 TA: OmniSub Sub-Accounts

The Transfer Agent manages the Alpaca OmniSub integration. Each investor gets a sub-account within the Liquidity omnibus account at Alpaca.

3.1 Data Model

Listing 1: OmniSubAccount struct

```
type OmniSubAccount struct {
    ID string // Internal record ID
    TenantID string // Org from JWT (X-Org-Id)
    CompanyID string // Issuer/company reference
    Name string // Account display name
    ExternalID string // Alpaca account ID
    Type string // "individual" or "entity"
    Status string // active | inactive | closed
    Metadata map[string]string // Custom fields
    CreatedAt time.Time
}
```

3.2 API Endpoints

Method	Path	Purpose
POST	`/v1/ta/omnisub/accounts`	Create sub-account (individual or entity). TenantID always overridden from JWT.
GET	`/v1/ta/omnisub/accounts`	List all sub-accounts for the authenticated org.
GET	`.../accounts/{id}`	Get a single sub-account. Cross-org blocked.
POST	`.../accounts/{id}/orders`	Place order. Requires symbol, side, qty.
GET	`.../accounts/{id}/positions`	Positions computed from filled orders.
GET	`.../accounts/{id}/tax`	Tax docs. Wired to Alpaca 1099 API.
POST	`.../corporate-actions`	Splits, dividends, mergers.

Table 1: TA OmniSub endpoints (`ta/omnisub_routes.go`).

3.3 Tenant Isolation

The `TenantID` field is *always* overridden from the `X-Org-Id` header (set by the gateway from the verified JWT). Client-supplied tenant values are ignored. Every list/get query is filtered by `org_id = ?` using parameterized queries to prevent injection.

4 BD: Multi-Provider Omnibus

The Broker-Dealer service manages funding, balance queries, crypto liquidation, and quoting across **16 broker/exchange providers**.

4.1 Provider Registry

Provider credentials are stored in the platform's `provider_configs` collection (encrypted at rest). The broker library provides a runtime registry with dynamic add/remove/toggle:

Provider	Capabilities
Alpaca	US equities + crypto, OmniSub, sandbox
Interactive Brokers	Global equities, options, futures, forex
BitGo	Institutional crypto custody, multi-sig
Binance	Crypto spot + derivatives
Kraken	Crypto spot, staking
Gemini	Crypto spot, custody
Coinbase	Crypto spot, Commerce (deposits)
SFOX	Crypto smart order routing
FalconX	Institutional crypto OTC
Fireblocks	Institutional custody, MPC wallets
Circle	USDC, stablecoin infrastructure
Tradier	US equities, options
Polygon	Market data
CurrencyCloud	FX, cross-border payments
LMAX	Institutional FX, crypto
Finix	Payment processing

Table 2: Supported broker/exchange providers.

4.2 Omnibus API Endpoints

Method	Path	Purpose
POST	`/v1/bd/omnibus/fund`	Fund sub-account via ACH. Auto-detects provider.
GET	`/v1/bd/omnibus/balance`	Sub-account balance. Optional `?provider=` hint.
POST	`/v1/bd/omnibus/liquidate`	Sell crypto for USD via best exchange.
GET	`/v1/bd/omnibus/quote`	Quotes from all providers. Returns best.

Table 3: BD Omnibus endpoints (`bd/omnibus.go`).

4.3 Provider Admin

Administrators manage provider credentials at runtime via `/v1/bd/admin/providers` (GET, POST, PUT, DELETE, toggle). Credentials are encrypted with per-tenant keys derived from HSM (AWS KMS / GCP Cloud KMS / Zymbit in production, env var in dev).

5 ATS: Market Data & Order Execution

The ATS serves 13,474 markets (13,401 US equities + 73 crypto) sourced from Alpaca's Data API with a two-layer cache:

- **Valkey** (shared): 30-second TTL on snapshots (prevents Alpaca 429s)
- **In-memory** (per-pod): Fallback when Valkey is unavailable

5.1 Market Data Endpoints

Method	Path	Purpose
GET	`/v1/assets`	Full market catalog with live prices
GET	`/v1/assets/{id}`	Asset detail + live quote
GET	`/v1/assets/{id}/quote`	Real-time bid/ask snapshot
GET	`/v1/assets/{id}/chart`	Historical OHLCV bars
GET	`/v1/assets/{id}/book`	Order book depth
GET	`/v1/quotes`	Batch quotes (`?symbols=...`)

Table 4: ATS market data endpoints (`ats/exchange.go`).

5.2 On-Chain Settlement

Trade settlement is **inline and blocking**:

1. CEX matching engine fills the order
2. ATS burns source token from seller's MPC wallet
3. ATS mints target token to buyer's MPC wallet
4. Settlement record written to `pending_settlements` (audit log)
5. If chain unavailable: trade flagged `settlement_failed`, retry cron every 30s

The Liquid EVM blockchain is the source of truth. SQLite is a read cache that is always re-derivable from chain state.

6 KYC/KYB Onboarding Pipeline

Account opening requires full KYC (individuals) or KYB (entities) before any sub-account is provisioned at Alpaca.

6.1 Account Types

Type	Category	Submission
Individual	Personal brokerage	KYC
Trust	Held by trustee	KYB
Corporate	C-Corp, S-Corp	KYB
LLC	Limited Liability Company	KYB
LP	Limited Partnership	KYB
Other	Non-profit, sole proprietorship	KYB

Table 5: Supported account types.

6.2 Onboarding Flow (5 Steps)

1. **Personal Info**: Name, email, phone, DOB, SSN (or EIN for entities)
2. **Address**: Residential + optional mailing address
3. **Employment**: Status, employer, annual income, net worth, source of funds
4. **Disclosures**: Control person, FINRA affiliation, PEP status
5. **Options Suitability**: Investment objective, experience, risk tolerance, time horizon

Each step validates via Zod schemas and saves a draft to AsyncStorage. Final submission goes to:

- `POST /compliance/kyc` for individuals
- `POST /compliance/kyb` for entities

6.3 Post-Approval Automation

On compliance approval, the following fires automatically:

1. AML screening via Jube (external sidecar) or in-process rules
2. Biometric enrollment via SecureGate (face capture + 1:N dedup)
3. Alpaca sub-account auto-provisioned (OmniSub)
4. Webhook events: `account.approved`, `kyc.approved`, `aml.cleared`
5. Email notification to investor

7 Alpaca Sandbox Provisioning

The CLI tool `cmd/alpaca-setup` batch-provisions sandbox accounts for development and testing environments.

Listing 2: Alpaca sandbox setup

```
# Create accounts from fixture data and fund each with $50K
go run ./cmd/alpaca-setup --env dev

# Preview without executing
go run ./cmd/alpaca-setup --env dev --dry-run
```

Per account, the tool:

1. Checks if the account already exists (by email query)
2. Creates the account with contact, identity, disclosures, and agreements
3. Creates an ACH relationship (sandbox auto-approves)
4. Transfers $50,000 (sandbox max per day; run twice for $100K)
5. Saves the email → account_id mapping for K8s deployment

Credentials are sourced from `ALPACA_API_KEY`/`ALPACA_API_SECRET` environment variables, or extracted from the K8s secret `ats-secrets`.

8 Identity Header Standard

All Liquidity services use a unified identity header convention:

Security: All identity headers are stripped from incoming requests before JWT validation. Headers are only set from verified JWT claims. Services never trust client-supplied identity headers.

Header	Description
`X-User-Id`	Authenticated user ID (from JWT `sub` claim)
`X-Org-Id`	Organization/tenant ID (from JWT `owner` claim)
`X-User-Email`	User email address (from JWT `email` claim)

Table 6: Canonical identity headers. Set by gateway after JWT validation.

9 Webhook Events

The BD fires 16 webhook event types to registered subscribers:

Event	Trigger
`account.created`	New onboarding application submitted
`account.approved`	Admin approves application
`account.rejected`	Admin rejects application
`kyc.submitted`	KYC data submitted (step 2 or 5)
`kyc.approved`	KYC verification passes
`kyc.rejected`	KYC verification fails
`aml.flagged`	AML screening returns flagged/blocked
`aml.cleared`	AML screening returns cleared
`compliance.alert`	Wallet screen returns blocked/high risk
`order.placed`	Order submitted to matching engine
`order.filled`	Order fully filled
`order.partial_fill`	Order partially filled
`order.canceled`	Order canceled
`transfer.initiated`	MPC settlement initiated
`transfer.completed`	MPC settlement confirmed on-chain
`account.suspended`	Account suspended (reserved)

Table 7: BD webhook event types.

Delivery: HMAC-SHA256 signed, async via durable task queue, 5 retries with exponential backoff, 15-second timeout per attempt.

10 Deployment

All three services deploy as StatefulSets (2 replicas) across three GKE clusters:

Env	GCP Project	API Domain	Branch
Devnet	`liquidity-devnet`	`api.dev.satschel.com`	`dev`
Testnet	`liquidity-testnet`	`api.test.satschel.com`	`test`
Mainnet	`liquidity-mainnet`	`api.main.satschel.com`	`main`

Table 8: Deployment environments.

Images pushed to GAR: `us-docker.pkg.dev/liquidity-registry/{type}/{name}:{branch}`.

11 Roadmap

1. **Position aggregation**: Compute real positions from filled orders (TA)
2. **Tax document API**: Wire Alpaca W-8BEN and 1099 endpoints (TA)
3. **Corporate actions**: Process real splits, dividends, mergers from Alpaca events (TA)
4. **Live Alpaca wiring**: Replace sandbox URLs with production Broker API
5. **Smart order routing**: Best-execution routing across all 16 providers
6. **FIX 4.4 protocol**: Institutional order submission via CEX
7. **MPC wallet creation**: Auto-provision 2-of-3 threshold wallets on account approval